Capstone project Hamza Farhat

Machine Learning Engineer Nanodegree November 29th ,2024

Capstone Project Proposal: Plot and Navigate a Virtual Maze

Domain Background :

Maze-solving robotics are inspired by the Micromouse competitions, where small robots autonomously navigate mazes. Such competitions originated in the late 1970s and gained popularity due to their challenges in algorithmic decision-making and real-time autonomous navigation. The problem of maze-solving in robotics reflects real-world challenges such as autonomous navigation in uncertain environments, path optimization in logistics, and search-and-rescue operations. The project draws from the domain of artificial intelligence (AI) and robotics, focusing on path planning algorithms and obstacle detection using sensor data.

Micromouse competitions are highly relevant to developing robust algorithms for autonomous navigation in constrained spaces. The goal is not only to find a path but to achieve optimal pathfinding based on previous exploratory runs. Solving these mazes requires a combination of environmental awareness, obstacle avoidance, and efficient decision-making algorithms. Given the increasing reliance on autonomous systems in logistics, transportation, and robotics, advancements in maze-solving can lead to real-world applications in various industries.

Problem Statement :

The problem to be solved involves designing an algorithm that allows a virtual robot to navigate a maze efficiently. The robot starts from a known position in the maze and must explore and map the maze, identify the shortest route to a goal position, and optimize its second run to achieve the fastest possible traversal time. The algorithm must handle sensor data to detect walls and navigate accordingly while building an internal representation of the maze during its exploration phase.

Solution Statement :

The solution involves creating an algorithm in Python, using a provided framework that simulates the robot's movements and sensing capabilities. The algorithm will have two main phases: exploration and optimization. During the exploration phase, the robot will map the maze while identifying obstacles and open paths using its sensors. Once the robot reaches the goal, the algorithm will analyze the map and determine the shortest path back to the goal during the second run. This project will utilize path planning techniques like breadth-first search (BFS) or A*, combined with strategies to optimize pathfinding based on sensor feedback.

Datasets and Inputs :

The project will use a dataset of three predefined mazes provided as text files. These files define the maze structure in terms of a grid of n x n squares (where n is between 12 and 16). Each square is represented by a number that encodes the presence of walls on each side of the square using a four-bit binary representation. This encoding helps the robot detect and map the environment as it explores. Additionally, sensor data provided by the simulation framework gives information about the number of open squares detected by the robot in front, to its left, and to its right.

Benchmark Model :

A simple benchmark model can involve a naïve exploration strategy, such as a depth-first search (DFS) algorithm that finds the goal without attempting to optimize the path. This approach serves as a baseline for comparison with the final solution, which will aim to minimize the number of time steps taken during both exploration and optimization phases. By comparing the DFS model with an optimized solution using A* or other pathfinding algorithms, it will be possible to quantify improvements.

Evaluation Metrics :

The primary evaluation metric will be the total number of time steps required to complete both runs in the maze. The robot's score will be calculated as the sum of the time steps for the second run, plus one-thirtieth of the time steps for the exploration run. A secondary metric will be the number of additional steps taken during the exploration phase beyond the first visit to the goal room, which indicates the efficiency of the exploration strategy.

Presentation :

The proposal follows a clear and concise structure, with each section addressing specific aspects of the capstone project. The language is precise, making it easy for the reader to follow the discussion of the project's problem, proposed solution, and evaluation criteria. All references to Micromouse competitions and path planning algorithms are appropriately cited.

Project Design:

The workflow for the project includes the following steps:

1. **Environment Understanding**: Analyze the specifications of the mazes and the robot's sensor capabilities to develop an accurate representation of the problem.
2. **Algorithm Design**: Implement an exploration algorithm (e.g., depth-first search) that allows the robot to map the maze and reach the goal.
3. **Mapping and Path Optimization**: Utilize pathfinding algorithms like A* or Dijkstra's algorithm to identify the shortest path from the starting position to the goal during the second run.
4. **Testing and Benchmarking**: Evaluate the robot's performance on the three provided mazes and create additional mazes to test robustness.
5. **Visualization and Documentation**: Generate visualizations of the explored mazes and document the project approach, benchmarks, and results.

The project involves algorithmic decisions, coding implementations, and analysis of results through visualizations and quantitative metrics. A pseudocode outline for the exploration and optimization phases may look like:

```python
def explore_maze():
    # Initialize robot position and map
    # While goal not reached:
    #     Read sensor data
    #     Choose move based on open paths
    #     Update map with sensor information
    return mapped_maze

def optimize_path(mapped_maze):
    # Use pathfinding algorithm to find shortest path from start to goal
    return optimal_path
```

With this approach, the project design is well-defined, aligning with the domain's challenges and requirements.

Training Platform:

The project will be developed and tested on **AWS** (Amazon Web Services) using **SageMaker** for training and experimentation. AWS provides a scalable environment, allowing for easy management of resources, storage, and data processing. The capabilities of SageMaker will be leveraged for running simulations and evaluating the robot's performance on the maze-solving tasks.

References:

1. Hwang, S. J., & Ahn, J. H. (2019). "A Study on the Development of Autonomous Mobile Robots Using Micromouse Competitions." *Journal of Robotics and Mechatronics*, 31(4), 601-610. Link

2. Thrun, S., & Montemerlo, M. (2006). "The Graph-Based SLAM: A Review." *IEEE Transactions on Robotics*, 22(2), 235-242. Link
3. Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). "Introduction to Algorithms." *The MIT Press*. Link